National Energy Research Scientific Computing Center (NERSC)

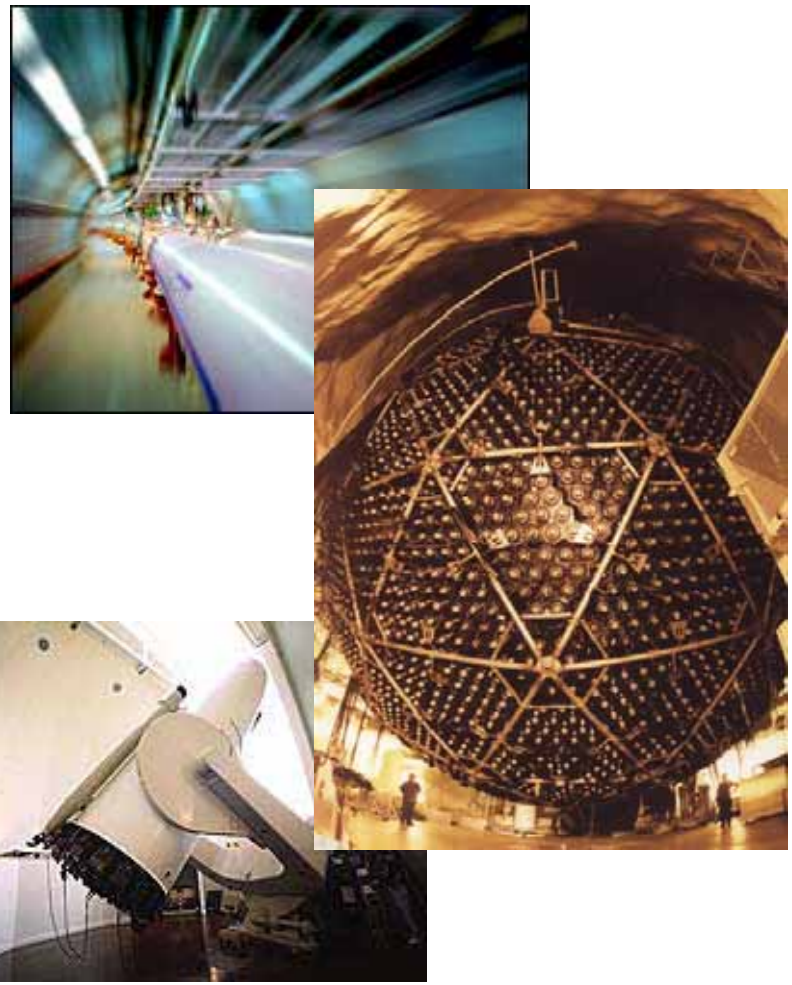
CHOS - CHROOT OS

Shane Canon
NERSC Center Division, LBNL
SC 2004
November 2004

Background

PDSF is a medium size cluster used by a diverse group of High Energy and Nuclear Physics Groups

- ATLAS
- CDF
- STAR
- KamLAND
- SNO
- SNFactory (Astrophysics)



Motivation

Problem

Groups were starting to request different versions of RedHat (RH 7.2, RH 7.3, RH8)

Solution

CHOS - In house developed framework for supporting multiple OSs concurrently on a single system.

Requirements

- Support multiple OSs concurrently on each node
- Not require partitioning the cluster
- Be nearly transparent to the users
- Integrate with the batch/scheduler system
- Easily deployable across the cluster
- Scale with the number of requested OS releases
- Must be secure

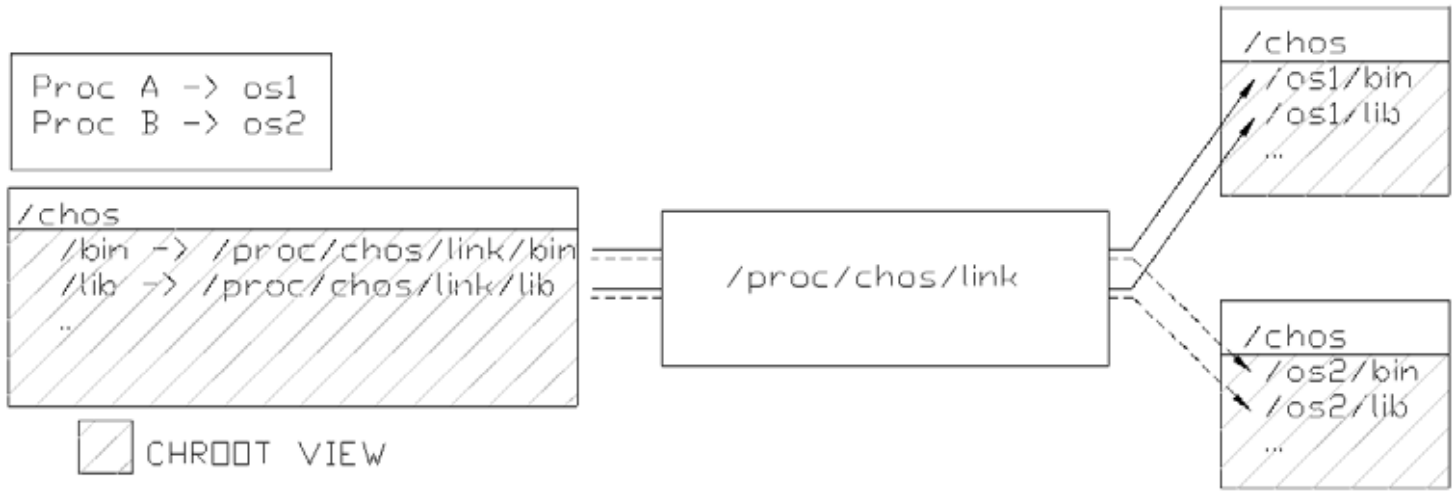
CHOS - CHROOT OS

- At its core, CHOS is chroot'ing into an alternate OS
- However, this alone isn't enough
  - File systems (both real and virtual)
  - Batch integration needed
  - Should be transparent and automatic
  - Preferred that it scaleable for many OSs

Kernel Module

- Creates to files in proc file system (/proc/chos)
  - /proc/chos/link - Special symbolic link
  - /proc/chos/setlink - Writable file to set path for link
- /proc/chos/link has the following traits
  - Settable by setlink
  - Each process sees link pointing to its set value
  - Child processes inherit value of parent
- Following checks
  - Only root can set valid paths

The link file



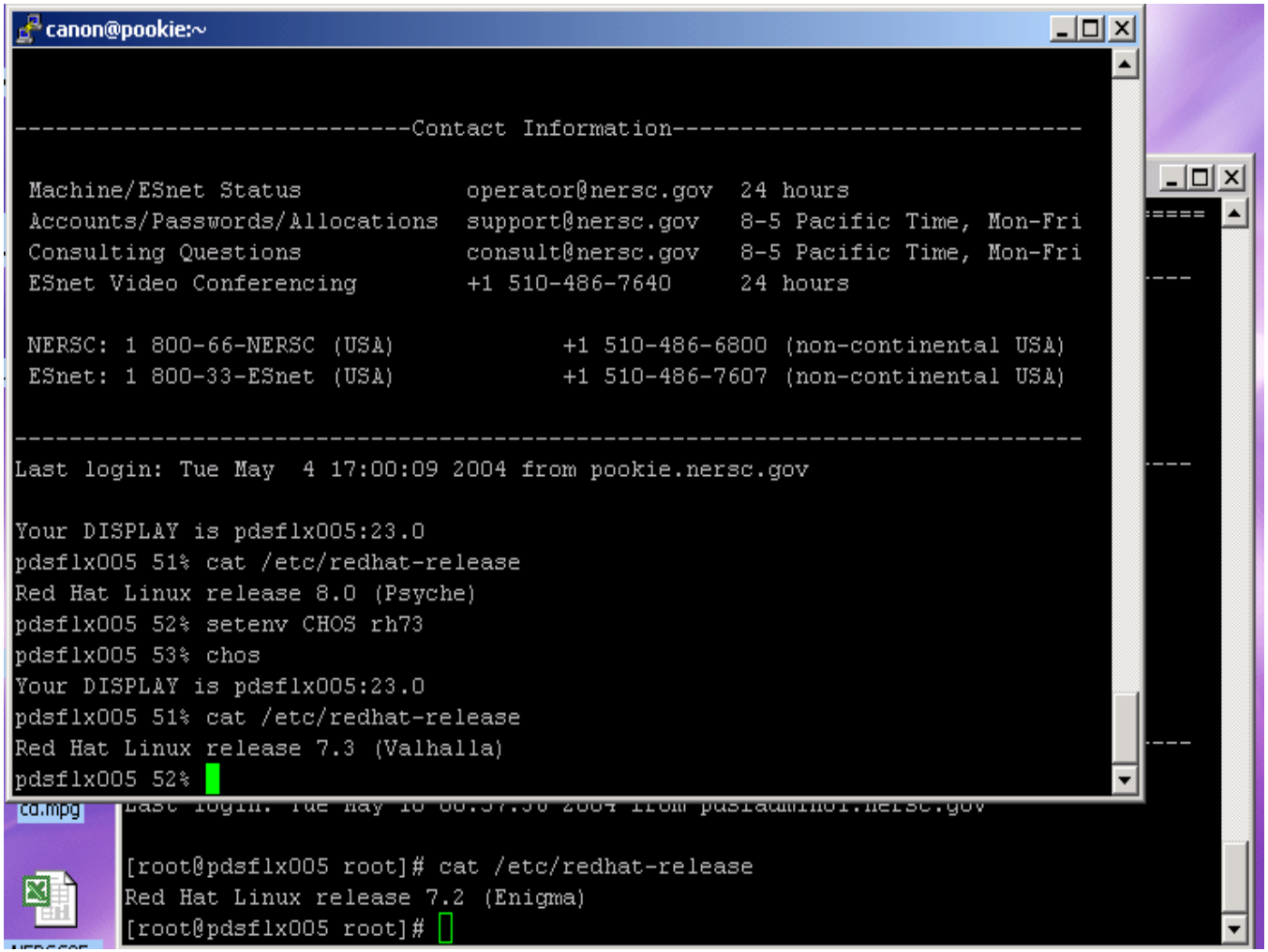
PAM Module

- PAM module that provide a “session” component
- PAM module looks at contents of .chos file in the user’s home directory
- Performs the necessary steps to initiate a CHOS session
- Sets CHOS environment variable
- Can be added to PAM configuration for ssh to automatically use the alternate OS upon login

Batch Integration

- Modified job starters are used for that batch system
- Job starter looks for CHOS environmental variable
- Automatically switches if CHOS variable is set to a valid OS
- PAM module sets CHOS variable, so no further action is required by the user wanting to run the same OS

CHOS – In Action



Use Examples

- Different groups can have their own custom OS
- Independently upgrading base OS without forcing users to switch platforms
- Provide test bed for users evaluating or migrating to new OSs.
- Support 32 bit OS on 64 bit base OS (and kernel)
- Provide access to older releases (un-maintained) in more secure fashion for re-running old codes or applications
- Run binaries compiled for a specific release in CHOS, while running other services in base OS

Security

- Services would typically be run out of just the base OS
- Disable setuid programs in alternate OSs to limit security risks. If application needs to be setuid, symlink to local installation
- CHROOT is a privileged operation for a reason
  - CHOS allows administrator to specify which alternate OSs are allowed
  - CHOS checks against this list before initiating a CHOS session

Current Status

- Tested with both 2.4 and 2.6 kernels
- Base OS: RedHat, SuSE, Fedora, Scientific Linux
- Alternate OS: RedHat, Fedora, Scientific Linux
- Tested with multiple versions of RedHat and SuSE

Future Work

- Simplified installation - Already in RPM format. Future release may automatically mount local file systems under CHOS
- PAM enabled job starter - Re-use PAM module for batch system as well. This job starter could have other uses (pam_limits).
- Kernel patch version instead of module to avoid some tricks

Conclusion

- Dealing with competing requirements from users is a typical problem for shared resources
- CHOS greatly diminishes this problem for providing various operating systems
- CHOS also helps decouple the needs of the system administrator from the needs of the user